**PhD Plus Data Literacy in R Cheatsheet**

**Set working directory**

setwd(“path/to/directory”)

Use tab key to drill into directory tree.

Use .. to go back up one branch.

Or Session…Set Working Directory

**Install/Update/Load Packages**

install.packages(“package”)

Or Tools…Install Packages…

library(package)

Tools…Check for Package Updates…

package::function indicates function in package

Example: readr::read\_csv()

**Assignment**

Use <- or =

Alt + - (Win) Option + - (Mac) to insert <-

**Import data**

d <- read.csv(“path/to/file.csv”)

Or using read\_csv() from readr package:

d <- read\_csv(“path/to/file.csv”)

Use readxl package to import Excel files.

Use haven package to import SAS, SPSS, Stata files.

**Glance at data frame named “d”**

View(d); names(d)

str(d); dplyr::glance(d)

summary(d); head(d); tail(d)

**Comparison and Logical**

== (equality)

!= (not equal)

>,>= (greater than, greater than or equal to)

<,<= (less than, less than or equal to)

& (and)

| (or)

! (not)

%in% (matching operator)

**Missing values**

Missing values indicated with NA

NA = not available

is.na() returns TRUE if value missing, FALSE otherwise

**Create/combine vectors**

x <- c(2, 4, 8)

y <- c(x, 10) # append 10 to 2,4,8

**TRUE/FALSE**

TRUE = 1, FALSE = 0

x <- c(2, 4, 8)

x > 3

[1] FALSE TRUE TRUE

sum(x > 3) # how many TRUE?

[1] 2

**Counts and proportions**

Count of males/females in column “sex” of data frame “d”:

table(d$sex)

Proportion of females

mean(d$sex == “female”)

If any missing values, set na.rm = TRUE

mean(d$sex == “female”,

na.rm = TRUE)

**Basic statistical functions**

mean(); median(); sd(); var()

quantile() # percentiles

length() # number of values (n)

sqrt() # square root

log() # natural log

log10() # log base 10

min(); max()

range() # min and max

**Tidyverse**

Collection of packages.

library(tidyverse) loads 8 packages.

readr: functions for importing data

dplyr: functions for data wrangling

ggplot2: visualization

tidyr: change shape of data frame

stringr: functions for manipulating text

tibble: “improved” data frames

forcats: functions for working with factors

purrr: functional programming tools

**Plotting with ggplot2**

<https://raw.githubusercontent.com/rstudio/cheatsheets/main/data-visualization.pdf>

Example data frame: d

|  |  |  |
| --- | --- | --- |
| x | y | g |
| 1300 | 3.8 | “a” |
| 1400 | 3.2 | “b” |
| 1280 | 2.9 | “a” |

library(ggplot2)

distribution of y

ggplot(d) + aes(x = y) +

geom\_histogram()

ggplot(d) + aes(x = y) +

geom\_density()

scatterplot of x and y

ggplot(d) + aes(x, y) +

geom\_point()

**Plotting with ggplot2 (cont’d)**

scatterplot of x and y conditional on g

ggplot(d) + aes(x, y) +

geom\_point() +

facet\_wrap(~g)

scatterplot of x and y points colored by g

ggplot(d) + aes(x, y, color=g) +

geom\_point()

scatterplot of x and y, semi-transparent points

ggplot(d) + aes(x, y, color=g) +

geom\_point(alpha = 1/5)

# alpha ranges from 0

# (invisible) to 1 (solid)

scatterplot with x and y and smooth trend line

ggplot(d) + aes(x, y) +

geom\_point() +

geom\_smooth()

# method=“lm” for straight line

distribution of y for each level of g

ggplot(d) + aes(x = g, y = y) +

geom\_boxplot()

ggplot(d) + aes(x = g, y = y) +

geom\_violin()

ggplot(d) + aes(x = g, y = y) +

geom\_jitter(width = 0.2,

height = 0)

Add title, axis labels, etc

ggplot(d) +

aes(x, y, color=g) +

geom\_point() +

labs(x = “SAT”, y = “GPA”,

title = “SAT vs GPA”)

**Basic data wrangling with dplyr**

<https://raw.githubusercontent.com/rstudio/cheatsheets/main/data-transformation.pdf>

Example data frame: d

|  |  |  |
| --- | --- | --- |
| x | y | g |
| 1300 | 3.8 | “a” |
| 1400 | 3.2 | “b” |
| 1280 | 2.9 | “a” |

dplyr functions work with pipes.

Insert pipe operator:

Ctrl+Shift+M (Win); Cmd+Shift+M (Mac)

dplyr always returns a tibble (data frame)

NOTE: Assign result to save transformation!

Extract rows that meet a condition

d %>% filter(x > 1300)

Arrange data by columns in ascending order

d %>% arrange(y)

Arrange data by columns in descending order

d %>% arrange(desc(y))

Select specific columns

d %>% select(y, g)

d %>% select(-x) # all but x

Two useful select helpers

d %>% select(starts\_with("p"))

d %>% select(-starts\_with("p"))

d %>% select(ends\_with("ing"))

Add a column and save result

d <- d %>%

mutate(z = x – mean(x))

Summaries for each group (eg, mean)

d %>%

group\_by(g) %>%

summarize(m = mean(y))

Count membership in group

d %>% count(g)

Rename columns and save result

d <- d %>% rename(SAT = x)

# new\_name = old\_name

Drop obs missing on a given variable

d %>% drop\_na(y)

Create an indicator variable using if\_else

# j = 1 if y = 4, else 0

d <- d %>% mutate(j =

if\_else(y==4,1,0))

Random sample of 20 observations

d %>% sample\_n(20)

Combining dplyr functions and saving result

nd <- d %>%

filter(x > 1000) %>%

group\_by(g) %>%

summarize(m = mean(y))

**Working with dates**

Use lubridate to format dates. Use m, d, y to create function. Dates stored as number of days since 1/1/70. Eg, to format dates of form May 2, 2021 in column “date” of data frame “d”

library(lubridate)

d <- d %>%

mutate(date = mdy(date))